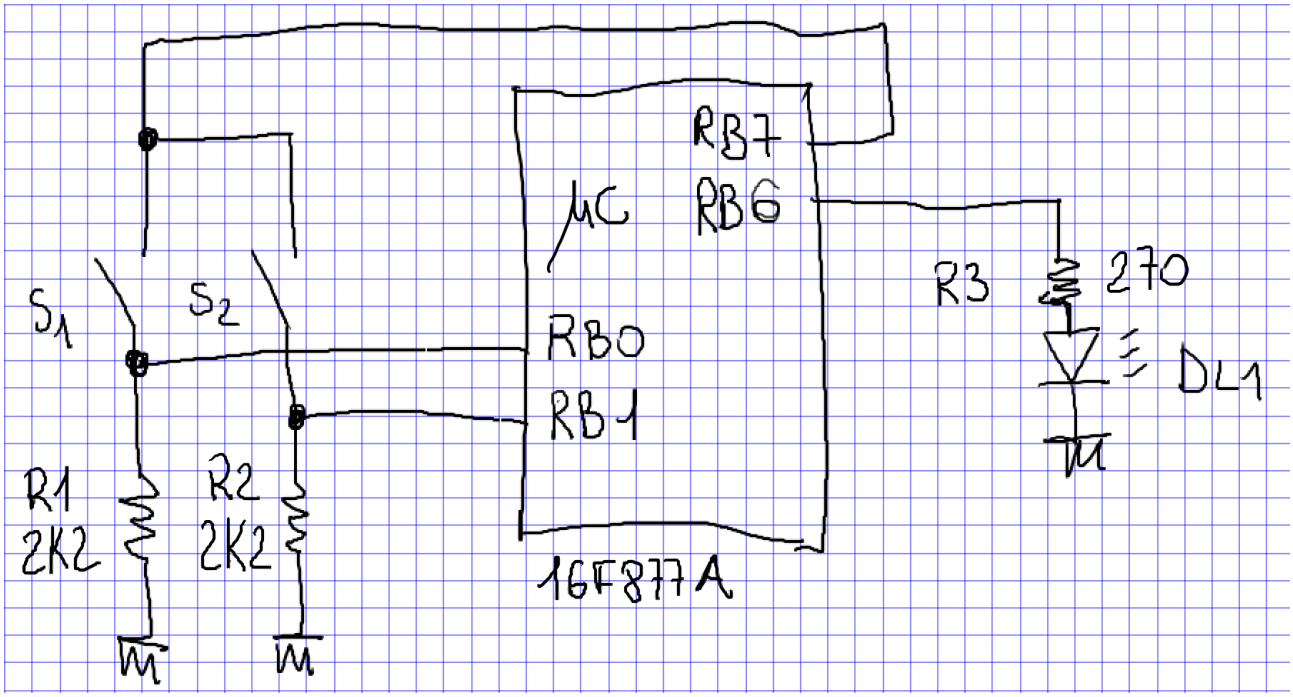
Dato lo schema seguente:



si richiede il seguente funzionamento:

S1 → S2 => DL1 OFF→ ON
S1 → S2 => DL1 ON→ OFF
Se premo in successione S1 poi S2 il led DL1 cambia lo stato, quindi se è spento si accende e se è acceso si spegne.

Vediamo un possibile codice in mikroC:

esempio 1 senza memorizzazione in eeprom

```
/*
==========================================================
=======================
     ESERCIZIO FATTO IN CLASSE
==========================================================
=======================*/
 // direttive del preprocessore, NON vanno chiuse con il ;
 #define PIPPO  0X10
 #define PULS_PREM 1
 #define PULS_RIL 0
 #define LED_ON 1

 // dichiarazione variabili globali
 short  cont;
 char pippo;
 char var1 = 0;

 // prototipi o definizioni delle funzioni : tipo  identificatore (tipo1, tipo2, ...)
 void configura (void);
```

```
 void interrupt (void);

void  interrupt ()
{
  // blocco istruzioni per la gestione dell'interrupt
}

 void configura (void)
 {
 // SETTAGGIO I/O  PER DEFAULT LI IMPOSTIAMO TUTTI COME INGRESSI
 TRISA = 0XFF;
 TRISB = 0X3F; // imposto gli ing e le uscite
 TRISC = 0XFF;
 TRISD = 0XFF;
 // SOLO I BIT 0, 1 E 2 SONO USATI PER LA PORTE
 TRISE.B0 = 1;
 TRISE.B1 = 1;
 TRISE.B2 = 1;
 // PORTA CON INGRESSI DIGITALI
 ADCON1 = 0X07;
 // OFF PULL-UP PORTB *** PER ATTIVARE LE R DI PULL-UP METTERE A "0" IL
BIT SEGUENTE
 OPTION_REG.B7 = 1;
 // altre istruzioni utili a configurazioni di registri legati alle periferiche interne e
I/O

 }

 void main {
 configura ();//chiama ed esegue la funzione che configura gli I/O e i registri
  // blocco istruzioni della funzione principale main
  while (1)
       {
  // scrivere qui il programma, all'interno del ciclo infinito

  PORTB.B7 = 1; //attivo l'uscita RB7 come riferimento dei pulsanti

  // leggo se premuto S1
  if(PORTB.BO == 1)
  {
  while(PORTB.BO == 1)
  {
  }
  Delay_ms(100);
  var1 = 1; // setto per informare che S1 è stato premuto
  }

  // leggo se premuto S2 e se è già stato premuto S1
  if((PORTB.B1 == 1) && (var1 == 1))
```

```
 {
 while(PORTB.B1 == 1)
 {
 }
 Delay_ms(100);
 var1 = 2;
 }
 // gestione led DL1
 if(var1==2)
 {
 if(PORTB.B6 ==0)
 {
 PORTB.B6 = 1; // accendo il led DL1
 var1 = 0;
 }
 else
 {
 PORTB.B6 = 0; // spengo il led DL1
 var1 = 0;
 }
 }
 }
}
```

esempio 2 con memorizzazione in eeprom

```
/*
==================================================
=======================
     ESERCIZIO FATTO IN CLASSE
==================================================
=======================*/
 // direttive del preprocessore, NON vanno chiuse con il ;
 #define PIPPO  0X10
 #define PULS_PREM 1
 #define PULS_RIL 0
 #define LED_ON 1

 // dichiarazione variabili globali
 short  cont;
 char pippo;
 char var1 = 0;

 // prototipi o definizioni delle funzioni : tipo  identificatore (tipo1, tipo2, ...)
 void configura (void);
 void interrupt (void);
 unsigned short Eeprom_Read (unsigned int);
 void Eeprom_write (unsigned int, unsigned int);
```

```
void  interrupt ()
{
  // blocco istruzioni per la gestione dell'interrupt
}

 void configura (void)
 {
 // SETTAGGIO I/O  PER DEFAULT LI IMPOSTIAMO TUTTI COME INGRESSI
 TRISA = 0XFF;
 TRISB = 0X3F; // imposto gli ing e le uscite
 TRISC = 0XFF;
 TRISD = 0XFF;
 // SOLO I BIT 0, 1 E 2 SONO USATI PER LA PORTE
 TRISE.B0 = 1;
 TRISE.B1 = 1;
 TRISE.B2 = 1;
 // PORTA CON INGRESSI DIGITALI
 ADCON1 = 0X07;
 // OFF PULL-UP PORTB *** PER ATTIVARE LE R DI PULL-UP METTERE A "0" IL
BIT SEGUENTE
 OPTION_REG.B7 = 1;
 // altre istruzioni utili a configurazioni di registri legati alle periferiche interne e
I/O

 }

 void main {
 configura ();//chiama ed esegue la funzione che configura gli I/O e i registri
  // blocco istruzioni della funzione principale main
  while (1)
       {
  // scrivere qui il programma, all'interno del ciclo infinito

  PORTB.B7 = 1; //attivo l'uscita RB7 come riferimento dei pulsanti

  // leggo se premuto S1
  if(PORTB.BO == 1)
  {
  while(PORTB.BO == 1)
  {
  }
  Delay_ms(100);
  var1 = 1; // setto per informare che S1 è stato premuto
  Eeprom_Write (1, var1); // memorizzo il valore di var1 nella loc. 1 della
eeprom
  }
```

```
var1 = Eeprom_Read (1); // leggo il valore della locazione 1 della eeprom
// leggo se premuto S2 e se è già stato premuto S1
if((PORTB.B1 == 1) && (var1 == 1))
{
while(PORTB.B1 == 1)
{
}
Delay_ms(100);
var1 = 2;
Eeprom_Write (1, var1); // memorizzo il valore di var1 nella loc. 1 della
eeprom
}
// gestione led DL1
if(var1==2)
{
if(PORTB.B6 ==0)
{
PORTB.B6 = 1; // accendo il led DL1
var1 = 0;
Eeprom_Write (1, var1);
}
else
{
PORTB.B6 = 0; // spengo il led DL1
var1 = 0;
Eeprom_Write (1, var1);
}
}
}
}
```